Ako testovať SQL injection a XSS útoky

SQL-injekcia a XSS (Cross-Site Scripting) sú dva najčastejšie útoky, ktoré zneužívajú špeciálne znaky v používateľskom vstupe:

**SQL-injekcia**

* **Čo to je?**  
  Útok, pri ktorom útočník vloží do vstupu časť SQL príkazu tak, aby zmenil vykonávanú databázovú operáciu.
* **Kľúčové znaky:**
  + **Single quote '** – používa sa na uzavretie reťazcov v SQL.
  + **Semikolon ;** – oddeluje viaceré SQL príkazy.
* **Príklad zraniteľného kódu (PHP):**
* $q = "SELECT \* FROM users WHERE username = '$userInput'";
* $result = $db->query($q);

Ak userInput = ' OR '1'='1, výsledný SQL je

SELECT \* FROM users WHERE username = '' OR '1'='1'

a vráti **všetkých** používateľov.

* **Ako testovať:**
  + Do povinného poľa (napr. meno) vložte ' OR '1'='1; očakávajte buď chybu, alebo žiadne neoprávnené údaje.
  + Vložte '; DROP TABLE users; -- a sledujte, či sa niečo vymaže.
  + Overte, že aplikácia nikdy nevracia surový SQL ani stack trace.
* **Ochrana:**
  + Používať **prepared statements** / parameterizované dotazy (napr. PDO s väzbou parametrov).
  + Na serveri vždy validovať a sanitizovať vstupy.

**XSS (Cross-Site Scripting)**

* **Čo to je?**  
  Útok, pri ktorom útočník doplní škodlivý JavaScript do vstupu, ktorý sa neskôr zobrazí iným používateľom.
* **Kľúčové znaky:**
  + **Angle brackets < >** – pre tagy (napr. <script>).
  + **Single a double quotes ' "** – na uzatváranie atribútov.
  + **Môže sa miešať aj so znakom ;** v JavaScripte.
* **Príklad zraniteľného kódu (JS templating):**
* <div>Vitaj, {{username}}</div>

Ak username = <script>alert('XSS')</script>, prehliadač vykoná kód.

* **Ako testovať:**
  + Vložte do textového poľa <script>alert('XSS')</script>; ak sa zobrazí alert, je zraniteľné.
  + Vyskúšajte varianty:
    - Vkladanie do atribútov: "><img src=x onerror=alert(1)>
    - HTML entity escaping: &lt;script&gt;…
  + Testujte aj v URL parametroch a lebels, ktoré sa dynamicky vkladajú do stránky.
* **Ochrana:**
  + **Escapovať** výstup: nikdy nenechať neohliadnutý vstup priamo v HTML.
  + Používať bezpečné knižnice (napr. OWASP ESAPI) alebo frameworky s automatickým escapingom (React, Angular…).
  + Nastaviť **Content Security Policy (CSP)** na obmedzenie spúšťania neautorizovaných skriptov.

Tým, že simuluješ zadanie týchto špeciálnych znakov a overíš, že sa namiesto nich pred odoslaním či zobrazovaním správne escapujú alebo viažu (parameterizujú), zabezpečíš formulár proti týmto typom útokov.